Beyond Positive/Negative Classification: Automatic Extraction of Sentiment Clues from Microblogs

Lu Chen
Kno.e.sis Center
Wright State University
Dayton, OH 45435 USA
chen@knoesis.org

Wenbo Wang
Kno.e.sis Center
Wright State University
Dayton, OH 45435 USA
wenbo@knoesis.org

Meenakshi Nagarajan
IBM Almaden Research Center
San Jose, CA 95120 USA
MeenaNagarajan@us.ibm.com

Shaojun Wang
Kno.e.sis Center
Wright State University
Dayton, OH 45435 USA
shaojun.wang@wright.edu

Amit P. Sheth
Kno.e.sis Center
Wright State University
Dayton, OH 45435 USA
amit@knoesis.org

ABSTRACT

Microblogging provides a large volume of text for learning and understanding people's sentiments on a variety of topics. Much of the current work on sentiment analysis of microblogs (e.g., tweets) focuses on document level polarity. However, identifying sentiment clues with respect to specific targets (e.g., named entities) can be more useful than pure document polarity results. For example, sentiment clues such as "*must see*", "*awesome*", "*rate 5 stars*" (in the movie domain) are much more meaningful than the polarities of tweets only. Previous attempts at single-word sentiment clue extraction from formal text will not suffice for extracting multi-word sentiment phrases. Single words "*must*" and "*see*" do not separately convey polarity, but their combination "*must see*" expresses strong positive sentiment towards a movie target. Another issue with identifying sentiment clues is identifying informal sentiment expressions, such as misspellings ("*kool*"), abbreviations ("*wtf*") and slangs ("*da bomb*"). In this paper, we propose an approach for automatically extracting both single-word and multi-word sentiment clues. Such clues can include both traditional and slang expressions. We also present a mechanism for assessing their target-specific polarities from an unlabeled microblog corpus. Our approach first leverages traditional and slang subjective lexicons to generate candidate sentiment clues given some specific target. It then incorporates inter-clue relations from corpora into an optimization model to estimate the probability of a clue denoting positive/negative sentiment. Experiments using microblog data sets on two different domains – movie and person – show that the proposed approach can effectively 1) extract single-word as well as phrase sentiment clues, 2) identify both traditional and slang sentiment clues, and 3) determine their target-specific polarities. We also demonstrate how the proposed approach is superior in comparison with several baseline methods.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Information filtering; I.2.7 [**Natural Language Processing**]: Text analysis

General Terms

Algorithms, Experimentation

Keywords

Sentiment Extraction, Sentiment Analysis, Opinion Mining, Optimization, Nonlinear programming

1. INTRODUCTION

Microblogging provides a convenient and instant way for people to share their sentiments on various topics anytime and anywhere. The ever growing microblogs offer a wealth of data that can be used for learning and understanding people's sentiment. Automatic sentiment analytics is a requirement if we want to assess the pulse of any population around any topic. There have been many studies [24–33] and applications[1] built on Twitter[2] to analyze sentiments in tweets. However, most existing research are more concerned with "***What*** *(is the sentiment polarity of a microblog?)*" instead of "***Why*** *(is a microblog having that sentiment polarity?)*." In other words, what are the sentiment clues causing the sentiment polarity? For example, considering users' opinions on the movie *Transformers 3*, besides reporting a relevant tweet as positive, it would be more meaningful to provide sentiment clues, such as "*must see*", "*awesome*", "*rate 5 stars*", etc. We define a "*sentiment clue*" as a single word or phrase that causes sentiment polarity on a target (i.e., user

[1]Such as TipTop(http://feeltiptop.com/),
Twitter Sentiment (http://twittersentiment.appspot.com/),
Tweetfeel (http://www.tweetfeel.com/),
Twitris (http://twitris.knoesis.org/), etc.
[2]http://twitter.com/

specified topic) in the text. While sentiment clues are important in understanding the polarity of text, their extraction from microblogs is quite challenging. Consider the following examples, in which we denote the target topic and the on-target sentiment clues in boldface and italic, respectively.

1. Saw the movie **Friends With Benefits**. So *predictable*! I *want my money back*.
2. Alright enough of **Taylor Swift**. She is *gud* but I am still *not a fan*.
3. **The King's Speech** was *bloody brilliant*. Colin Firth and Geoffrey Rush were fantastic!

The first observation is that a sentiment clue may not necessarily be a single word. Phrases can convey sentiment. Identifying which expression causes the sentiment is a hard problem. In example 1, none of single words ("*want, my, money, back*") or parts of the expression ("*want my, my money, etc.*") alone causes the sentiment, however, the phrase "*want my money back*" as a whole indicates negative sentiment. Secondly, informal expressions like misspellings, abbreviations and slangs, which are common in microblogs, can be sentiment clues too. "*gud*" in example 2 is such a case. Furthermore, sentiment clue extraction should be with respect to the target of concern. On the one hand, the extracted sentiment clues should be on-target. As in example 3, since the target is **The King's Speech**, "*bloody brilliant*" that refers to the target should be identified, while "*fantastic*" should be ignored because it acts on a different topic. On the other hand, the polarity assigned to the clue should be target-specific. For example, "*predictable*" could indicate positive sentiment with respect to stocks (e.g., *the predictable stock to own for the long term*), however, in example 1, it should be assigned a negative polarity referring to its target – movie **Friends With Benefits**.

The difficulties of sentiment clue extraction are well understood in the community, and many attempts have been undertaken across various types of documents. Some of these include reviews [3, 5–7, 9, 12–17], news articles [2, 4, 8, 18–20], blog posts [21] and general web documents [10, 11, 22]. Most previous research attempt to extract single-word sentiment clues and lack support for identification of sentiment phrases. Moreover, since informal language usage is not an issue for these datasets, little attention has been given to identification of slang sentiment clues. In our approach, we extract single-word and phrase-level sentiment clues, traditional and slang sentiment clues, and assess their target-specific polarities from microblogs.

Our approach consists of two main steps. In the first step, a large set of candidate sentiment clues is obtained. Based on the assumption that any sentiment clue contains at least one subjective word, a comprehensive subjective lexicon is constructed by collecting subjective words from both traditional (e.g., MPQA[3]) and slang (e.g., Urban Dictionary[4]) lexical resources. On-target subjective words in microblogs are spotted using this lexicon, and candidate sentiment clues containing at least one such word are added to the candidate set. In the following step, sentiment clues are identified from the candidate set and their polarities are assessed. Since the polarities of subjective words in the lexicons may be different for different topics, we do not apply the polarities inherited from lexicon resources directly. Instead, we apply an optimization model to estimate *polarity probabilities* of candidate clues according to their inter-clue relations (discussed in Section 3.2.1) in the corpus.

To validate the performance of our approach, we conduct experiments on microblog data sets from Twitter on two different domains – movie and person. The data sets contain 168K microblogs talking about movies, and 258K microblogs talking about persons. The results show that our approach can effectively extract single sentiment words as well as multi-word sentiment phrases, identify both traditional and slang sentiment clues and determine their target-specific polarities. To quantitatively measure our approach, we compare it with several baselines. In both tasks of sentiment clue extraction and sentiment classification of microblogs using the extracted clues, our approach outperforms the baseline methods. And with the increasing sizes of corpora, the advantage of our approach is more prominent.

The remainder of the paper is organized as follows. We discuss the related work in Section 2. In Section 3, we present our approach. The experiments and results are described in Section 4, followed by conclusion and future work in Section 5.

2. RELATED WORK

Sentiment analysis is one of the most popular topics of investigation. The authors of [1] present a survey which covers a broad range of issues and provides an in-depth review of the techniques and approaches in this area. Here, we focus on literature that is relevant to our task, including word/phrase level sentiment clue extraction, and sentiment analysis of microblogs.

2.1 Sentiment Clue Extraction

Extraction of sentiment clues has been explored in many studies, either as the main task (e.g., lexicon construction) or as a subtask of sentence or document level sentiment analysis. Research in [2–7,10–17,19] are some examples of efforts made in this area. Extracting single-word sentiment clues and assessing their polarities have been broadly supported, and here we are more concerned with the following two issues: (1) dealing with multi-word sentiment phrases, and (2) identifying slang clues.

Relatively fewer efforts have focused on the task of extracting phrase-level sentiment clues. Turney [3] manually defines a few part-of-speech (POS) patterns of two-word phrases (e.g., "JJ JJ" refers to two consecutive adjectives) to extract sentiment phrases, and estimates polarities of extracted phrases using point-wise mutual information by querying a search engine. Some studies [14–16] identify the boundaries of multi-word phrases through syntactic parsing, and the polarity of a whole phrase relies on that of its component words from a predefined sentiment lexicon in a pattern based manner. The work in [8] also employs a sentiment lexicon of words with given polarities and applies a supervised method

[3]http://www.cs.pitt.edu/mpqa/

[4]http://www.urbandictionary.com/

to identify contextual polarity of manually annotated sentiment phrases. Our approach is different from the above mentioned approaches in the following ways. We learn the target-specific polarity of sentiment clues from the corpus and do not depend on the generic prior polarity of words from a sentiment lexicon. In addition, we do not reply on POS tagging and syntactic parsing to identify the boundary of a phrase, because such approach is less effective in informal language context (e.g., microblogs). Our treatment of phrases is similar to [10] in that we also extract n-grams as candidates. However, we extract sentiment clues with respect to a specific target, and assess their target-specific polarities, while the target of sentiment is not a concern for the work in [10].

Some studies explore the issue of identifying slang expressions. For example, the work in [10] extracts sentiment phrases without using language-dependent resources such as POS tagging or WordNet, thus, the results are not limited to expressions of specific classes (e.g., an adjective occurring in Wordnet), but can contain formal expressions as well as slang, misspellings, etc. The work in [23] uses a sentiment lexicon which is built upon Urban Dictionary to identify sentiment words (especially informal words, such as slang) from user comments. We also exploit Urban Dictionary for identifying candidate slang clues. However, the polarity of each clue is not determined by polarities of related words from Urban Dictionary. Instead we apply an optimization model which leverages the information implicit in the corpus to estimate target-specific polarities of clues. Moreover, the previous approaches do not take target into consideration.

2.2 Sentiment Analysis of Microblogs

To the best of our knowledge, extraction of sentiment clues have not been studied on microblogs. Studies in [24]- [33] follow the supervised approach for document level sentiment classification of microblogs. Besides manual annotation, most work in the literature use different ways to obtain training data. For example, the work in [25] obtains labeled data from a few sentiment detection websites over Twitter. The work in [29–32] leverage the hashtags and emoticons in microblog messages for building training data. The work in [33], the authors first use a lexicon-based method to perform sentiment classification with high precision and then a supervised classifier is applied to improve the recall using the training examples provided by the previous lexicon-based approach. The work in [24] takes the sentiment target into consideration, and classifies tweets according to whether they contain positive, negative or neutral sentiments about a given target. While the above supervised techniques focus on the question: "*what is the sentiment polarity of a tweet*", our unsupervised approach tries to answer the question: "*what are the sentiment clues causing the polarities?*".

3. THE PROPOSED APPROACH

Let Δ be a collection of microblogs with respect to target T. Without loss of generality, T can be an individual topic or a set of topics of the same type in domain of interest (e.g., T can be a specific restaurant/movie, or a set of restaurants/movies). We consider T as a topic or a group of topics in the same domain so that the extracted clue can be assigned a fixed polarity with respect to the topic or all topics of T. It is widely recognized that the sentiments expressed by words/phrases are domain dependent, for example, "*predictable*" is negative in the movie domain while it is positive in finance and economics domain. Spotting the target mentioned in text is not the focus of this paper; so we assume that the targets have been marked in the microblogs of the corpus Δ. Our objective is to extract sentiment clues which cause the positive or negative sentiments on target T in microblogs from Δ, and assign each extracted clue its target-specific polarity (i.e., positive or negative).

To solve this problem, the proposed approach includes two main steps. In the first step, candidate sentiment clues are collected from Δ (refer to Section 3.1). In the second step, sentiment clues are identified from these candidates and each of them is assigned its polarity (refer to Section 3.2). As we discussed earlier, the key difficulties encountered include: (1) identifying the on-target clues, (2) dealing with multi-word phrases, (3) dealing with informal expressions, (4) assessing the target-specific polarity of the clues. Our approach leverages traditional and slang lexical resources as well as local and global information from corpora to handle these issues.

3.1 Extraction of Candidate Sentiment Clues

One widely adopted approach to narrow down the number of candidate clues is to select words/phrases belonging to a certain pattern, e.g., adjectives or an adjective followed by a noun. However, this approach is less effective in dealing with microblogs, in which the informal nature of sentence construction poses considerable difficulties for POS taggers. Moreover, there are diverse forms of expressions in microblogs, which cannot be fully captured by a few predefined patterns.

Our approach is not restricted to words/phrases with any specific part-of-speech patterns (e.g., adjective), because we do not want to miss out high recall in the case of any POS errors. Alternatively, we list all n-grams (up to length five) which contain at least one subjective word as candidates. The intuition behind the approach is that a sentiment clue usually contains at least one subjective[5] word, e.g., "want" in "want my money back", and "must" in "must see". The reason to list all n-grams is to generate as many candidates as possible, to increase the recall, and leave the job of selecting true sentiment clues to the optimization algorithm (refer to Section 3.2). It's worthwhile to mention that the purpose of applying subjective lexicons is to reduce the search space of candidate clues. In fact, we do not rely on the polarity from existing lexicons to infer target-specific polarity. In Section 3.1.1, we discuss how we exploit existing lexical resources to build a comprehensive subjective lexicon, which is then used to generate candidate n-gram clues described in Section 3.1.2.

3.1.1 Subjective Lexicon Construction

The task here is to build a comprehensive subjective lexicon containing both traditional and slang words. A *subjective lexicon* L is a dictionary of subjective words, in which each subjective word is assigned the polarity (i.e., positive or negative in the binary case) it expresses in a general sense.

[5]"subjective" is the opposite of "objective," and sentiment expressions are always "subjective."

Many lexical resources can be used for building L. One such resource is SentiWordNet[6], in which each synset of WordNet is assigned three sentiment scores: *PosScore*, *NegScore* and *ObjScore* according to its positivity, negativity and objectivity. We also use the MPQA subjective lexicon, which contains 8221 subjective words, each assigned a prior polarity. Another resource incorporated is the General Inquirer[7], from which we only use the 1915 positive words from the *Positiv* category and 2291 negative words from the *Negativ* category.

While the above mentioned resources provide a wealth of traditional subjective words, they are not sufficient for dealing with social content like microblogs in which slang expressions are very common. Hence it is necessary for L to incorporate slang subjective words. Urban Dictionary (UD) is a popular online slang dictionary with definitions written and voted by users. In addition to the glossary definitions, each word defined in this dictionary is associated with a list of related words to interpret the word itself. For example, the word "*rockin*" has the following related words in UD: "*awesome, cool, sweet, rock, rocking, amazing, hot, etc.*". However, UD does not specify the subjectivity or polarity of each word. To obtain subjective slangs and assess their polarities from UD, we have designed an algorithm using only a small set of seed sentiment words. These seed words along with their polarities were manually identified and regarded as target independent (i.e., the polarity of each word is fixed). For example, the word "excellent" is contained in the seed set since it is always positive no matter what the target is. This seed set is denoted as S. The algorithm starts by putting all words of the seed set S into a query set Q.

For a word w in the query set Q, the algorithm queries UD to obtain the list of related words for w. The first ten related words in the list along with the word w itself are treated as a "document." A frequency matrix is created to record the count of the co-occurrence of any pair of words in any document. This matrix is updated with every new obtained document. For example, assuming a new document d is obtained for the word ($w = rockin$), where $d = \{$rockin, awesome, cool, sweet, rock, rocking, amazing, hot, $\ldots\}$, then for each pair of words contained in d, add 1 to its corresponding element in the frequency matrix. The word ("rockin") is removed from the query set, and all its related words in that document which have never been added to the query set are added to the query set. This process continues until the query set becomes empty.

As the next step, the polarity of each word in the frequency matrix is determined and the subjective words are selected. The general idea is that a word is positive/negative if it is strongly associated with positive/negative words. Here the strength of association between two words is measured by the frequency with which they co-occur according to the information in the frequency matrix. For each word in the matrix, select the five subjective words (according to SentiWordNet, MPQA and GI) which most frequently co-occur with it. If more than three out of five associated words are positive or negative, the word is positive or negative. For example, according to this algorithm, the strongest associated subjective words for "*rockin*" are "*amazing, sexy, sweet, great, awesome*". All of these words are positive according to the lexical resources. Hence the word "*rockin*" is identified as positive and added to subjective lexicon L. Using this algorithm, a total of 3521 slang words were identified as subjective words from UD. Together with words from SentiWordNet[8], MPQA and GI, the subjective lexicon L contains 13606 words, in which 4315 words are positive, 8721 words are negative, and the remained are annotated as neutral. Note that the polarity information in L is not used for generation of candidate sentiment clues, but will be used as a different way to initialize input values for the optimization algorithm for comparison purpose in Section 4.

[6]http://sentiwordnet.isti.cnr.it/

[7]http://www.wjh.harvard.edu/ inquirer/

3.1.2 Candidate Sentiment Clue Generation

Our goal in this step is to extract all the candidate sentiment clues with respect to target T from corpus Δ. Our idea is to first identify an on-target **root** word, based on which we generate candidate clues. A root word has to satisfy the following two conditions: a) it is a subjective word according to the subjective lexicon L. b) it acts on target T. For each microblog in Δ, we use SentParBreaker[9] to do the sentence splitting and parse each sentence using Stanford Parser[10] to get the typed dependencies between words. After stemming[11], we spot all the subjective words in the microblogs of Δ based on the subjective lexicon L. The dependency relation between a subjective word and the target T is used as one way to judge whether the word acts on target T. Unlike many other studies, which limit dependency relations to some specific types (e.g., *mod* for modifier), we relax dependency relations to any type of dependency to avoid missing proper candidates due to informal language usage. We also use proximity to determine whether a word is on-target to improve robustness in recognizing subjective words associated with a target. In other words, a subjective word which is near to the target (within four words distance) can also be selected.

All n-grams up to length five which contain a root word are extracted as the candidate clues. A candidate clue will be removed if its first or last word is a stop word (e.g., "*love it*" is removed because "*it*" is a stop word). However, if a stop word only occurs in the middle of a clue, we choose to keep the clue (e.g., "want my money back" will not be removed, although "my" is a stop word). In addition, candidate clues containing a conjunction (e.g., "*predictable and boring*") or punctuation will be removed too. These removed candidates are actually redundant, e.g., "*is a must see*" is not necessary since there is "*must see*", and "*predictable and boring*" can be covered by "*predictable*" and "*boring*".

3.2 Assessment of Polarity Probability

After generating candidate sentiment clues, the problem becomes identifying the sentiment clues which cause positive

[8]We only use the words with the *PosScore* or *NegScore* higher than 0.75, or the difference between *PosScore* and *NegScore* higher than 0.50 from SentiWordNet.

[9]http://text0.mib.man.ac.uk:8080/scottpiao/sent_detector

[10]http://nlp.stanford.edu/software/lex-parser.shtml

[11]WordnetStemmer (http://projects.csail.mit.edu/jwi/api/edu/mit/jwi/morph/WordnetStemmer.html) is used to get the stems of words.

or negative sentiment with respect to target T in corpus Δ. We propose to transform this problem into the problem of estimating its polarity (positive/negative) probabilities. The main idea is that a positive/negative sentiment clue will have high positive/negative probability. If there is not much difference between the positive and negative probability of a candidate, the candidate tends to be neutral or objective. We are concerned only with positive and negative clues and do not consider the case of identifying neutral expressions. Formally, for each candidate clue c_i, we define its *positive probability* $Pr^{pos}(c_i)$ as the probability that c_i indicates *positive* sentiment, and its *negative probability* $Pr^{neg}(c_i)$ as the probability that c_i indicates *negative* sentiment. Both positive and negative probabilities can be termed polarity probability. To model the intuition that the more likely c_i is positive/negative, the less likely it is negative/positive, we assume:

$$Pr^{pos}(c_i) + Pr^{neg}(c_i) = 1 \tag{1}$$

For example, a candidate with positive probability of 0.9, and negative probability of 0.1 is highly likely to be a positive clue (i.e. express positive sentiment). A candidate having its positive and negative probability as 0.45 and 0.55 respectively tends to be neutral or objective and should be filtered out.

As discussed in Section 1, the assessed polarity probability of each clue should be target-specific. This assessment cannot rely on any general language resources (e.g., general sentiment lexicons). We propose to learn consistency/inconsistency relations between clues (i.e., inter-clue relations) from the corpus, and use this information as input for an optimization model to predict polarity probability of each candidate clue. Because the candidate clues are on-target, and the assessment of their polarity probabilities are based on their relations in the corpus, we are able to predict their target-specific polarities using this approach. In Section 3.2.1, we describe how we extract inter-clue relations from corpus Δ, and discuss the optimization model to predict the polarity probability for each candidate clue as shown in Section 3.2.2.

3.2.1 Inter-Clue Relation Extraction

We exploit two types of inter-clue relations: consistency relation and inconsistency relation, denoting whether the sentiments of two clues are consistent or inconsistent in the context with each other. Formally, We use *consistency network* $N^{cons}(P, R^{cons})$ to model consistency relations, where P is a set of clue pairs, and R^{cons} is a set of weighted edges connecting each pair of clues. The weight of the edge in R^{cons} is the frequency that two clues of the pair appear as consistent sentiment clues in Δ. In a similar manner, we define *inconsistency network* $N^{incons}(P, R^{incons})$.

For each pair of two adjacent clues c_i and c_j in one microblog, they are identified as expressing *inconsistent* sentiment if (1) c_i is a part of c_j (but not equal to c_j), and c_j starts with a *negation* expression and ends with c_i, or (2) c_i appears before c_j (without overlap between them), where there is no extra[12] negation applied to c_i or c_j, and they are connected by *contrasting conjunctions* (e.g., but, though, although, nevertheless, etc.) The algorithm uses a list of negation and contrasting expressions for this task. On the other hand, a pair of two adjacent clues c_i and c_j are identified as expressing *consistent* sentiment if c_i appears before c_j (without overlap between them), where there is no extra negation applied to c_i or c_j and no contrasting conjunction connecting them. Note that only the relations of two adjacent clues (i.e., there is no other clues between them) are considered in the algorithm, since the relations of two clues are too vague to be confirmed if there are other clues between them.

For example, in the microblog "Alright enough of Taylor Swift. She is *gud* but I am still *not a fan.*", "*fan*" and "*not a fan*" have a inconsistency relation according to rule (1), and "*gud*" and "*not a fan*" share a inconsistency relation according to rule (2). While "*predictable*" and "*want my money back*' share a consistency relation in the microblog "Saw the movie Friends With Benefits. So *predictable*! I *want my money back.*"

In the above described manner, the algorithm extracts all consistency and inconsistency relations from Δ. Network N^{cons} is generated by connecting each pair of candidate clues with a weighted edge, of which the weight is the number of consistency relations between the pair of clues. If there is no consistency relation between two clues, their edge has weight 0. Network N^{incons} can be built in the similar way.

The reader must note that the consistency and inconsistency relations suggest the relations of target specific polarities between candidate clues. Based on these relations, we are able to estimate the target specific polarity of each clue. In the previous example, "*predictable*" and "*want my money back*" share a consistency relation with respect to a movie target. This relation suggests that "*predictable*" should have the same polarity as "*want my money back*", i.e., both of them are negative. With a different target such as a given stock, "*predictable*" is likely to have consistency relations with positive expressions. An optimization model is constructed based on this target specific information to estimate the polarity probability of each clue.

3.2.2 An Optimization Model

We formulate the problem of assigning positive probability $Pr^{pos}(c_i)$ and/or negative probability $Pr^{neg}(c_i)$ to a candidate clue c_i as an optimization problem, in which we obtain optimal positive/negative probability by minimizing an objective function. We first describe the objective function and then identify true positive/negative clues based on estimated positive/negative probability.

Given a pair of sentiment clues c_i and c_j, we define their *consistency probability* $Pr^{cons}(c_i, c_j)$ as the probability that c_i and c_j have the consistent sentiment (both positive or both negative). Based on the assumption of independence between events that c_i and c_j on being positive or negative, we get

$$\begin{aligned} Pr^{cons}(c_i, c_j) = Pr^{pos}(c_i)Pr^{pos}(c_j) + \\ Pr^{neg}(c_i)Pr^{neg}(c_j) \end{aligned} \tag{2}$$

Similarly, their *inconsistency probability* $Pr^{incons}(c_i, c_j)$ is

[12] The "extra" means the negation is not a part of c_j or c_i. It rules out the relation between "*gud*" and "*fan*".

defined as the probability that c_i and c_j indicate inconsistent sentiment (one being positive and the other being negative):

$$Pr^{incons}(c_i, c_j) = Pr^{pos}(c_i)Pr^{neg}(c_j) + Pr^{neg}(c_i)Pr^{pos}(c_j) \quad (3)$$

A consistency relation between two clues c_i and c_j suggests that c_i and c_j have consistent polarity, i.e. the expectation of their consistency probability is 1.0. The difference between the consistency probability and its expectation can be measured by squared error: $(1.0 - Pr^{cons}(c_i, c_j))^2$. Similarly, for an inconsistency relation, the corresponding difference is measured using $(1.0 - Pr^{incons}(c_i, c_j))^2$. The sum of differences for all the relations in two networks can be obtained by the sum of squared errors (SSE)[13]:

$$SSE = \sum_{i=1}^{n-1}\sum_{j>i}^{n}(w_{ij}^{cons}(1.0 - Pr^{cons}(c_i, c_j))^2 + w_{ij}^{incons}(1.0 - Pr^{incons}(c_i, c_j))^2) \quad (4)$$

where w_{ij}^{cons} and w_{ij}^{incons} are the weights of edges between c_j and c_j in N^{cons} and N^{incons}, respectively, and n is the total number of candidate clues. Note that the squared error (instead of absolute error) is employed so that the two kinds of relations cannot cancel each other. After substituting consistency and inconsistency probabilities in equation 4 from equations 2 and 3, we get the objective function:

$$SSE = \sum_{i=1}^{n-1}\sum_{j>i}^{n}(w_{ij}^{cons}(1 - (Pr^{pos}(c_i)Pr^{pos}(c_j) + Pr^{neg}(c_i)Pr^{neg}(c_j)))^2 + w_{ij}^{incons}(1 - (Pr^{pos}(c_i)Pr^{neg}(c_j) + Pr^{neg}(c_i)Pr^{pos}(c_j)))^2) \quad (5)$$

The estimated optimal positive and negative probabilities should be able to minimize the above objective function, so that the corresponding consistency and inconsistency probabilities will be closest to their expectations suggested by the networks. We represent the negative probability of c_i as $1 - Pr^{pos}(c_i)$ [14] in equation 5. To make the objective function more clear, we abbreviate $Pr^{pos}(c_i)$ and $Pr^{pos}(c_j)$ to x_i and x_j. Finally, we get the constrained optimization problem:

$$minimize \sum_{i=1}^{n-1}\sum_{j>i}^{n}(w_{ij}^{cons}(x_i + x_j - 2x_ix_j)^2 + w_{ij}^{incons}(1 - x_i - x_j + 2x_ix_j)^2)\} \quad (6)$$

subject to,

$$0 \leq x_i \leq 1, \qquad for \quad i = 1, \cdots, n$$

We collected a set of seed words in Section 3.1.1. Since their polarities do not change for most topics (e.g., “excellent”), there is no need to learn these polarities from corpus. Instead, we use these polarities as input to the optimization model. The positive probability of a positive seed word is 1.0, and the positive probability of a negative seed word is 0.0.

To solve this minimization problem with simple bounds, we choose to use L-BFGS-B[15] algorithm. This algorithm is based on the gradient projection method [34] to determine a set of active constraints at each iteration, and uses a limited memory BFGS matrix to approximate the Hessian of the objective function. It was shown in [35] that L-BFGS-B takes advantage of the form of the limited memory approximation to implement the algorithm efficiently. The initial guess for the parameters (the positive probabilities of clues) are needed as the input of L-BFGS-B algorithm, two ways of initializing the parameters are implemented and compared (see Section 4).

As a result of solving the minimization problem, we get the positive and negative probabilities of each candidate clue. Only those with high positive/negative probabilities are selected as positive/negative clues. Specifically, clues with positive/negative probability higher than threshold τ are identified as positive/negative clues. We use $\tau = 0.6$ in this work, other clues falling below the threshold are removed as noise. However, there might still be some noise among high polarity probability clues, the main reason being that the assessment of the polarity probability of some clues is based on sparse data. In the extreme case, a candidate, which only appears once in the corpus and happens to be represented as consistent with a positive clue, could be assigned a high positive probability. Such cases occur more often on long phrases due to their rare occurrence. For dealing with this problem, we use another score to measure the confidence of the estimation. For each extracted clue c_i, the score is calculated as: $\varepsilon = \frac{Pr^{pos}(c_i)*df(c_i)}{n_{words}(c_i)}$, where $df(c_i)$ is document frequency (the number of microblogs containing c_i), and $n_{words}(c_i)$ is the number of words it contains. c_i is removed from the result if its ε is less than threshold σ. We set $\sigma = 0.6$ in this work.

4. EXPERIMENTS

We now describe experiments validating the performance of our approach. Section 4.1 describes the experimental setup. In Section 4.2, we examine the quality of extracted sentiment clues by our algorithm in comparison with several baseline methods, and then demonstrate that the outcome of sentiment clue extraction can benefit other sentiment analysis applications, such as sentiment classification. We also conduct experiments to investigate the performance of our approach and other baselines with various sizes of corpora in Section 4.3.

4.1 Experimental Setup

We use two collections of microblogs (tweets) from Twitter: one contains 168,005 tweets about movies (movie domain), and the other contains 258,655 tweets about persons (person domain). Each tweet of the two collections contains either a movie or a person as the target.

Gold Standard: Since it is too costly to create the gold standard using all 426, 660 tweets from two collections, we created gold standard using 1500 tweets randomly sampled from the collection for each domain (totally 3000 tweets for both domains). The 3000 tweets were given to two groups of

[13]Note that the relations are symmetric, so j is limited to $j > i$ to avoid duplication.

[14]According to equation 1

[15]http://www.mini.pw.edu.pl/~mkobos/programs/lbfgsb_wrapper/index.html

Table 1: Distribution of Sentiment Clues

Clue Length	1	2	3	4	5	>5	Total
Movie Domain							
# of Positive Clues	103	42	15	8	5	4	177
# of Negative Clues	59	18	16	11	9	5	118
Person Domain							
# of Positive Clues	142	34	11	5	2	0	194
# of Negative Clues	56	14	9	0	2	1	82

Table 2: Distribution of Sentiment Categories

Category	Pos.	Neg.	Neut.	Obj.	Total
Movie Domain					
# of tweets	420	96	36	948	1500
Person Domain					
# of tweets	292	105	7	1096	1500

human annotators, and each group consists of three annotators. One group of annotators identified the positive and the negative sentiment clues which express sentiment towards the specified target from each tweet. The other group of annotators labeled each tweet as *positive*, *negative*, *neutral* and *objective* according to the overall sentiment toward the target.

We selected sentiment clues which are agreed by at least two annotators, and get 295 and 276 clues in movie and person domains, respectively. Table 1 shows the distribution of positive/negative clues. The percentages of multi-word phrases among all clues are 45.1% $(1-\frac{103+59}{177+118})$ in movie domain and 28.3% $(1-\frac{142+56}{194+82})$ in person domain, which underscores the importance of extracting sentiment phrases. We also get the 1500 tweets for each domain labelled with their sentiment categories. Table 2 illustrates the distribution of tweets belonging to different sentiment categories.

Baseline Methods: Our method is to extract sentiment clues with respect to specific target from unlabeled corpus. Under this setting, the following baselines were chosen for comparison:

MPQA: For each extracted root (on-topic sentiment word, see Section 3.1.2), simply look up its polarity (or its stem's polarity if the word itself is not in the lexicon) in the general-purpose subjective lexicon MPQA.
GI: In the same manner as MPQA, but here the General Inquirer is used instead of MPQA.
SWN: In the same manner as MPQA, but here the SentiWordNet is used instead of MPQA.
PROP: The propagation method is described in [5] . Starting with some seed sentiment words (here we apply the same seed set used in our algorithm), this method extracts new sentiment words and sentiment targets (topics or features) through a double propagation process. It uses a set of extraction rules based on different relations between sentiment words and targets, and also sentiment words and targets themselves. In our setting, sentiment targets have been specified, so we adapt the method to extract only sentiment words. In addition, the original method only concerns adjectives, so we extend it to extract adjectives, verbs, adverbs and nouns by relaxing the constraints of extraction rules, because these types of words are considered as sentiment carriers and supported by other methods in comparison.

Prior approaches which support phrases extraction [3,10,14–16] either lack consideration of target (e.g., [3, 10]) or need extra effort to develop patterns (e.g., [14–16]). Thus, we do not employ them here.

Our method is represented as "**COM**" (Constrained Optimization Model). As we mentioned before, we initialize the positive probabilities of candidate clues (the input of the L-BFGS-B algorithm) in two ways:
COM-const: assign constant 0.5 to all candidate clues as their initial positive probabilities.
COM-gelex: The initial positive probability of each candidate clue is set to 1.0/0.0 if the clue is positve/negative regarding the subjective lexicon L, otherwise 0.5.

Evaluation Measurement: we measure the quality of extracted sentiment clues using precision, recall and F-measure.

$$\begin{aligned} precision &= \frac{N_{agree}}{N_{result}} \\ recall &= \frac{N_{cover}}{N_{gold}} \\ F-measure &= \frac{2 \times precision \times recall}{precision + recall} \end{aligned}$$

where N_{agree} is the number of extracted clues which are consistent with the gold standard, N_{result} is the number of extracted clues, N_{cover} is the number of clues in the gold standard which are consistent with the extraction result, and N_{gold} is the number of clues in gold standard. Whether a clue is agreed or not is decided using "contain rule", thus, positive clue "good" is consistent with positive clue "pretty good" or vice versa. We also deal with the negation, thus, positive clue "good" is consistent with negative clue "not good" or vice versa. Since the baseline methods only extract single words, using "contain rule" is a fair way to measure and compare between those baselines and the proposed method. Note that N_{agree} and N_{cover} may not be equal, since there could be multiple extracted clues consistent with one clue in gold standard, or multiple clues in gold standard consistent with one clue in the extraction result.

4.2 Quality of the Extracted Sentiment Clues

Table 3 shows the precision, recall and F-measure on evaluating 1,500 tweets of gold standard for both movie and person domains. We can see that both versions of our method, COM-const and COM-gelex, outperform the baseline methods in both domains. To be more specific, our best F-measure in movie domain is 8%-21% higher than that of baselines, and in person domain, our best F-measure is 8%-25% higher than that of baselines. In both domains, the highest precision is achieved by COM-const, the highest recall is achieved by COM-gelex, and the two highest F-measures are achieved by both of our methods. Among all of the three lexicon-based methods (MPQA, GI and SWN), MPQA provides the best result, however, its precision is

Table 3: Results of Sentiment Clue Extraction

Method	Precision	Recall	F-measure
Movie Domain			
MPQA	0.3542	0.5136	0.4193
GI	0.3318	0.4320	0.3753
SWN	0.2876	0.4898	0.3624
PROP	0.4742	0.5034	0.4884
COM-const	**0.6433**	0.5170	**0.5733**
COM-gelex	0.5164	**0.5578**	0.5363
Person Domain			
MPQA	0.3523	0.4746	0.4045
GI	0.2949	0.4058	0.3416
SWN	0.2161	0.3659	0.2718
PROP	0.5352	0.3696	0.4372
COM-const	**0.5879**	0.4710	**0.5230**
COM-gelex	0.4599	**0.5507**	0.5012

relatively low. The PROP method performs quite well in terms of precision, but it suffers from low recall, especially in person domain.

Compared with the lexicon-based methods, our method can get significantly higher precision for sentiment clue extraction. This indicates that the sentiment of many clues are target-specific, which can be captured by our method. The lexicon-based methods do not use the information in the corpus and they cannot handle the target-specific polarity of clues. Compared with the PROP method, which applies syntactic relation rules to extract clues and determine their polarities, our method can get much higher recall. One reason could be that the informal language usage in microblogs brings much difficulty to syntactic parsing, and depending on the parsing results, it likely results in missing many sentiment clues. Our method reduces the dependence on the language parsing by using subjective lexicon.

In many sentiment analysis applications, the extracted clues are used for the task of sentiment classification. Thus, we also apply the sentiment clues extracted by different methods for sentiment classification of microblogs, as another way of evaluation. To classify the microblogs as *positive*, *negative*, *neutral* or *objective*, we follow an unsupervised method using the set of extracted clues collected previously. For each microblog, we identify the clues in the set as features. Since we only concern ourselves with the sentiment towards the specified target, those features which do not contain any on-target word are removed. We use the method described in Section 3.1.2 (i.e., using typed dependencies and proximity) to determine whether a word acts on the target. In addition, we remove features which are parts of other features, e.g., "good" is removed if "not good" is also a feature (Note that because the baseline methods only have single word features, they are not affected). We also deal with negation for the baseline methods because they cannot handle it directly. In this manner, we get the features (sentiment clues) of the microblog, and each feature is assigned a score (i.e., 1 for positive feature and -1 for negative feature). The scores

Table 4: Results of Microblog Sentiment Classification Using the Extracted Clues

Method	Precision	Recall	F-measure
Movie Domain			
MPQA	0.6566	0.5507	0.5990
GI	0.6381	0.4982	0.5595
SWN	0.5266	0.5018	0.5139
PROP	0.7677	0.5507	0.6413
COM-const	**0.8015**	0.5851	**0.6764**
COM-gelex	0.7164	**0.5905**	0.6474
Person Domain			
MPQA	0.5250	0.3639	0.4299
GI	0.4419	0.3292	0.3773
SWN	0.2979	0.3119	0.3047
PROP	0.5371	0.3045	0.3887
COM-const	**0.6351**	0.3317	0.4358
COM-gelex	0.5925	**0.3886**	**0.4694**

of all the features are added together to get the sentiment polarity of the microblog (i.e., *positive* with the sum > 0, *negative* with the sum < 0, and *neutral* otherwise). If no feature can be identified from the document, it is labeled as *objective*.

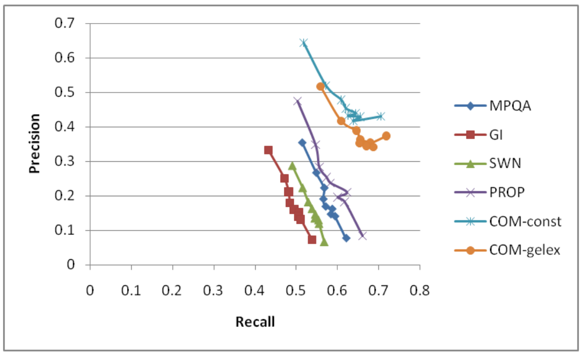
We use precision, recall and F-measure to measure the result of sentiment classification, and count only three sentiment categories (i.e., *positive*, *negative* and *neutral*). The results on both domain are shown in Table 4. This evaluation shows that the performance of different methods is quite consistent with the quality of the sentiment clues they extracted. Our method achieves the best F-measure on both domains.

4.3 Effect of Varying Corpora Sizes

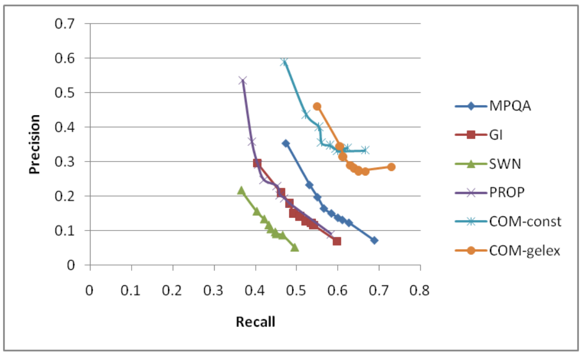
We also conduct experiments to investigate the effect of corpora size on the quality of extracted clues. We expect to get higher quality results as more inter-clue relations are learned from larger corpora. We evaluated all approaches over corpora of sizes from 1,500 to 48,000. Because it is not practical to manually label such large amount of tweets, we compare results extracted from corpora of different sizes against the same 1,500 tweets of gold dataset. To make the comparison meaningful, we make sure that all corpora of different sizes include the 1,500 tweets of gold dataset.

Figure 1 shows how precision, recall and F-measure change as we increase the sizes of corpora for both domains. Note that the computed precision may be worse than the true quality obtainable using a larger corpus, since the small gold standard includes only a limited number of sentiment clues. To gain more insights into the results, we use both precision-recall curve and F-measure to examine the relative performance of different methods.

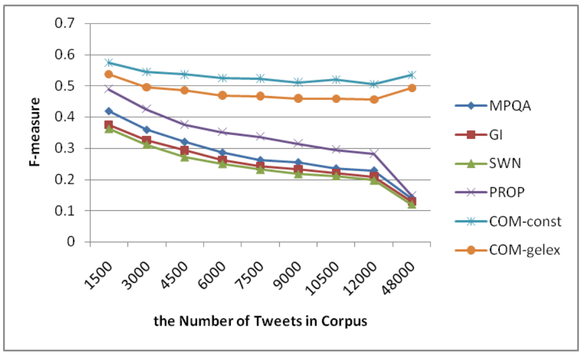
Figures 1a and 1b show our methods outperform the baselines. Specifically, COM-const tends to get the highest precision and COM-gelex tends to get the highest recall. Among all the baselines, PROP works best for the movie data, es-



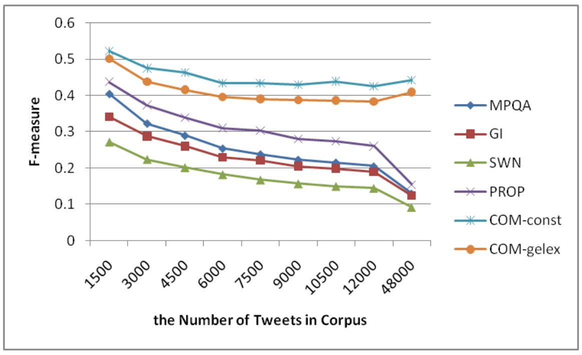
(a) Recall-precision Curve (Movie Domain)



(b) Recall-precision Curve (Person Domain)



(c) Average F-measure (Movie Domain)



(d) Average F-measure (Person Domain)

Figure 1: Results of all approaches with various corpora sizes

pecially on the recall aspect, while MPQA provides the best results for person domain. However, all baseline methods suffer from sharp decline of precision with the increasing recall. By manually checking the results generated by these methods, we find many irrelevant words that actually do not indicate sentiment on their own with respect to the target. Our approach can effectively filter these noises because we estimate polarity probabilities from a global point of view by constructing relation networks out of many tweets. Note that both versions of our approach make increases on both precision and recall when we increase the size of corpora from 12,000 (the second right most point of each line) to 48,000 (the right most point of each line). It suggests that our method could benefit from more relations extracted from larger corpora.

Observing from Figures 1c and 1d, in both domains, we can see that both COM-const and COM-gelex makes significant improvement on F-measure over four baselines, and COM-const provides best results. F-measures of our approach decline a little as we increase the corpora sizes (<= 6,000). However, the F-measure maintains at the same level or decrease very slightly until the size of corpus reaches 12000. From size 12000 to 48000, the F-measure even goes up a little. This can be explained in the same way as the changes of precision and recall with increasing sizes of corpora.

Table 5 illustrates a small sample of extracted sentiment clues by our method (with a corpus of 48000 tweets in movie domain). Most of these clues are not identified by the baselines. For both positive and negative categories, we present clues with their target specific polarities, e.g., “bomb” (positive), “predictable” (negative), etc. We also present clues which appear as multi-word phrases: e.g., “must see” “thumbs down”, etc. Finally, we show slang sentiment clues which are quite popular in microblogs, e.g., “luv”, “stoopid”, etc. These slang clues can be identified by our approach because our subjective lexicon includes slang words from Urban Dictionary. These concrete examples show that the proposed method performs in an intuitively satisfactory manner.

5. CONCLUSION AND FUTURE WORK

In this paper, we present an approach for extracting sentiment clues with respect to a specified target from unlabeled microblog corpus. Our approach addresses three shortcomings of current approaches, (1) extraction of multi-word sentiment clues, (2) identification of slang expressions, and (3) assessment of target-specific polarities of sentiment clues. To be able to identify multi-word sentiment clues, n-grams instead of only single words are extracted as candidate clues. For dealing with slang expressions, Urban Dictionary is exploited to enrich our subjective lexicon with slang words. Based on this lexicon, candidate clues including slang expressions are identified. To assess the target-specific polarities of clues, on-target candidate clues are identified using dependency analysis and proximity, and inter-clue relations from the corpus are incorporated in to an optimization model to estimate the polarities of clues. Using a variety of tweets from two popular domains we find that we are able to extract single-word and multi-word sentiment clues, identify both traditional and slang sentiment clues, and predict polarity with respect to the target. We also demonstrate how

Table 5: Sample Sentiment Clues Extracted by the Proposed Method

Positive			Negative		
Target-specific	**Phrase**	**Slang**	**Target-specific**	**Phrase**	**Slang**
bomb	must see	aight	average	thumbs down	stoopid
intense	eye candy	tight	sleepy	screwed up	craptastic
kick ass	rate 5 stars	rad	predictable	nothing special	rediculous
light-hearted	funny as hell	luv	copying	pretty lame	wacky
pretty crazy	pretty damn funny	awsome	cheapest	not even funny	superbad
cried alot	better than i expected	kool	little slow	sucked big time	dense
rules box office	even better the second time	rockin	little long	Don't waste your money	crapest

the proposed approach is superior to several baseline methods, in terms of both quality and scalability with respect to the size of the corpora.

In this study, we employed simple heuristics including the frequency and the length of candidates to identify sentiment clues and filter noise. This filtering may eliminate rarely used or long sentiment clues. We are currently exploring more sophisticated heuristics to identify high quality sentiment clues and reduce noise.

6. REFERENCES

[1] B. Pang and L. Lee. *Opinion mining and sentiment analysis.* Foundations and Trends in Information Retrieval, 2008.

[2] E. Riloff, J. Wiebe and T. Wilson. Learning subjective nouns using extraction pattern bootstrapping.In Proceedings of CoNLL, 2003.

[3] P. Turney. *Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews.* Proceedings of ACL, 2002.

[4] E. Breck, Y. Choi and C. Cardie. *Identifying expressions of opinion in context.* In Proceedings of IJCAI, 2007.

[5] G. Qiu, B. Liu, J. Bu and C. Chen. *Expanding domain sentiment lexicon through double propagation.* In Proceedings of IJCAI, 2009.

[6] Y. Lu, M. Castellanos, U. Dayal and C. Zhai. *Automatic construction of a context-aware sentiment lexicon: an optimization approach.* Proceedings of WWW, 2011.

[7] X. Ding, B. Liu and P. Yu. *A holistic lexicon-based approach to opinion mining.* In Proceedings of WSDM, 2008.

[8] T. Wilson, J, Wiebe and P. Hoffmann. *Recognizing contextual polarity in phrase-level sentiment analysis.* In Proceedings of HLT/EMNLP, 2005.

[9] W. Du, S. Tan, X. Cheng and X. Yun. *Adapting information bottleneck method for automatic construction of domain-oriented sentiment lexicon.* Proceedings of WSDM, 2010.

[10] L. Velikovich, S. Blair-Goldensohn, K. Hannan and R. McDonald. *The viability of web-derived polarity lexicons.* In Proceedings of HLT/NAACL, 2010.

[11] N. Kobayashi, K. Inui, Y. Matsumoto, K. Tateishi, and T. Fukushima. *Collecting evaluative expressions for opinion extraction.* In Proceedings of IJCNLP, 2004.

[12] A. Popescu and O. Etzioni. *Extracting product features and opinions from reviews.* In Proceedings of HLT/EMNLP, 2005.

[13] M. Hu and B. Liu. *Mining and summarizing customer reviews.* In Proceedings of SIGKDD, 2004.

[14] H. Kanayama, T. Nasukawa. Fully automatic lexicon expansion for domain-oriented sentiment analysis. In Proceedings of EMNLP, 2006.

[15] K. Nigam and M. Hurst. *Towards a robust metric of polarity.* In J. Shanahan, Y. Qu and J. Wiebe (Eds.). Computing Attitude and Affect in Text: Theory and Applications. Springer, 2006.

[16] J. Yi, T. Nasukawa, R. Bunescu, and W. Niblack. *Sentiment analyzer: Extracting sentiments about a given topic using natural language processing techniques.* In Proceedings of ICDM, 2003.

[17] A. Fahrni and M. Klenner. *Old wine or warm beer: Target-specific sentiment analysis of adjectives.* In Proceedings of the Symposium on Affective Language in Human and Machine, AISB 2008 Convention, 2008.

[18] T. Nasukawa and J. Yi. *Sentiment analysis: Capturing favorability using natural language processing.* In Proceedings of K-CAP, 2003.

[19] Y. Choi, Y. Kim and S. Myaeng. *Domain-specific sentiment analysis using contextual feature generation.* In Proceeding of CIKM workshop on Topic-sentiment analysis for mass opinion (TSA), 2009.

[20] Y. Choi, C. Cardie. *Adapting a polarity lexicon using integer linear programming for domain-specific sentiment classification.* Proceedings of EMNLP, 2009.

[21] V. Jijkoun, M. Rijke and W. Weerkamp. *Generating focused topic-specific sentiment lexicons.* In Proceedings of ACL, 2010.

[22] N. Kaji and M. Kitsuregawa. *Building lexicon for sentiment analysis from massive collection of HTML documents.* In Proceedings of EMNLP-CoNLL, 2007.

[23] D. Gruhl, M. Nagarajan, J. Pieper, C. Robson and A. Sheth. *Multimodal social intelligence in a real-time dashboard system.* The VLDB Journal, v.19 n.6, 2010.

[24] L.Jiang, M.Yu, M.Zhou, X.Liu, and T.Zhao. *Target-dependent Twitter sentiment classiïňĄcation.* In Proceedings of ACL, 2011.

[25] L. Barbosa and J. Feng. *Robust sentiment detection on Twitter from biased and noisy data.* In Proceedings of COLING, 2010.

[26] A. Agarwal, B. Xie, I. Vovsha, O. Rambow and R. Passonneau. *Sentiment analysis of Twitter data.* In Proceedings of ACL Workshop on Language in Social Media (LSM), 2011.

[27] A. Bermingham and A. Smeaton. *Classifying sentiment in microblogs: is brevity an advantage?.* In Proceedings of CIKM, 2010.

[28] G. Li, S. Hoi, K. Chang and R. Jain. *Micro-blogging sentiment detection by collaborative online Learning.* In Proceedings of ICDM, 2010.

[29] E. Kouloumpis, T. Wilson and J. Moore. *Twitter sentiment analysis: the good the bad and the OMG*! In Proceedings of ICWSM, 2010.

[30] A. Pak and P. Paroubek. *Twitter as a corpus for sentiment analysis and opinion mining.* In Proceedings of LREC, 2010.

[31] D. Davidov, O. Tsur and A. Rappoport. *Enhanced sentiment learning using Twitter hashtags and smileys.* In Proceedings of COLING, 2010.

[32] A. Go, R. Bhayani and L. Huang. *Twitter sentiment classification using distant supervision.* Technical report, Stanford Digital Library Technologies Project, 2009.

[33] L. Zhang, R. Ghosh, M. Dekhil, M. Hsu and B. Liu. *Combining lexicon-based and learning-based methods for Twitter sentiment analysis.* Hewlett-Packard Labs Technical Report, HPL-2011-89, 2011.

[34] D. Bertsekas. *Nonlinear Programming, 2nd Edition.* Athena Scientific, 1999.

[35] R. H. Byrd, P. Lu and J. Nocedal. *A limited memory algorithm for bound constrained optimization.* SIAM Journal on Scientific and Statistical Computing, 1995.